대 한 민 국 특 허 청
KOREAN INTELLECTUAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Intellectual Property Office.

출 원 번 호 : 10-2003-0034168
Application Number

출 원 년 월 일 : 2003년 05월 28일
Date of Application MAY 28, 2003

출 원 인 : 주식회사 하이닉스반도체
Applicant(s) Hynix Semiconductor Inc.

2003 년 06 월 30 일

특 허 청 장
COMMISSIONER

【서지사항】

【서류명】	특허출원서
【권리구분】	특허
【수신처】	특허청장
【제출일자】	2003.05.28
【발명의 명칭】	가속화 모드를 구비한 레지스터 제어 지연고정루프
【발명의 영문명칭】	Register controlled delay locked loop having acceleration mode
【출원인】	
【명칭】	주식회사 하이닉스반도체
【출원인코드】	1-1998-004569-8
【대리인】	
【명칭】	특허법인 신성
【대리인코드】	9-2000-100004-8
【지정된변리사】	변리사 정지원, 변리사 원석희, 변리사 박해천
【포괄위임등록번호】	2000-049307-2
【발명자】	
【성명의 국문표기】	곽종태
【성명의 영문표기】	KWAK,Jong Tae
【주민등록번호】	721105-1922915
【우편번호】	467-850
【주소】	경기도 이천시 대월면 사동리 현대전자아파트 101-1003
【국적】	KR
【발명자】	
【성명의 국문표기】	이성훈
【성명의 영문표기】	LEE,Seong Hoon
【주민등록번호】	671122-1117429
【우편번호】	467-860
【주소】	경기도 이천시 부발읍 신하리 559-3 청구아파트 105-803
【국적】	KR
【심사청구】	청구
【취지】	특허법 제42조의 규정에 의한 출원, 특허법 제60조의 규정에 의한 출원심사 를 청구합니다. 대리인 특허법인 신성 (인)

【수수료】

【기본출원료】	20	면	29,000 원
【가산출원료】	18	면	18,000 원
【우선권주장료】	0	건	0 원
【심사청구료】	8	항	365,000 원
【합계】	412,000 원		

【첨부서류】 1. 요약서·명세서(도면)_1통

【요약서】

【요약】

본 발명은 반도체 회로 기술에 관한 것으로, 특히 지연고정루프(delay locked loop, DLL)에 관한 것이며, 더 자세히는 레지스터 제어 DLL(register controlled DLL)에 관한 것이다. 본 발명은 소자의 동작 속도 증가에 따른 정확도 측면에서의 손실을 개선할 수 있는 가속화 모드를 구비한 레지스터 제어 지연고정루프를 제공하는데 그 목적이 있다. 본 발명은 DLL의 가속화 모드의 단위 지연량 'N ×unit_delay'의 N값을 최대 동작 주파수에 맞춰 고정시켜 사용하지 않고 동작 주파수에 따라 유동적으로 사용한다. 즉, 저주파 동작시에는 N값을 키워 가속화 모드의 단위 지연량을 증가시키고, 고주파 동작시에는 N값을 줄여 가속화 모드의 단위 지연량을 줄인다. 이 경우, 저주파 동작이든 고주파 동작이든 가속화 모드로 동작하는 사이클은 거의 비슷하지만 가속화 모드의 단위 지연량이 서로 달라져서, 결국 전체 사이클 측면에서는 지연고정 시간이 비슷해진다. 문제는 동작 주파수를 어떻게 파악하여 그에 대응하는 N값을 결정하는가 인데, 예컨대 동기식 반도체 메모리 소자의 경우에는 동작 주파수와 밀접한 관계가 있는 카스 레이턴시(CAS Latency), 모드 레지스터 세팅 등의 인덱스를 이용하면 된다.

【대표도】

도 9

【색인어】

레지스터 제어 지연고정루프, 가속화 모드, 딜레이 점프, 동작 주파수

【명세서】

【발명의 명칭】

가속화 모드를 구비한 레지스터 제어 지연고정루프{Register controlled delay locked loop having acceleration mode}

【도면의 간단한 설명】

도 1은 종래기술에 따른 DDR SDRAM의 레지스터 제어 DLL의 블록 다이어그램.

도 2는 가속화 모드를 구비한 종래기술에 따른 DDR SDRAM의 레지스터 제어 DLL의 블록 다이어그램.

도 3은 상기 도 2의 레지스터 DLL의 동작 파형도.

도 4는 대한민국 특허출원 제2002-66443호에 따른 가속화 모드를 구비한 DDR SDRAM의 레지스터 제어 DLL의 블록 다이어그램.

도 5는 상기 도 4의 레지스터 제어 DLL의 동작 파형을 예시한 도면.

도 6은 상기 도 5의 쉬프트 레지스터 및 제2 지연 라인의 회로 구성을 예시한 도면

도 7a 및 도 7b는 가속화 동작 모드가 종료되는 경우의 제1 및 제2 위상 비교기의 파형을 예시한 도면.

도 8은 상기 도 4의 지연부의 지연량(N ×unit_delay)을 결정하는 'N'값의 범위를 설명하기 위한 도면.

도 9는 본 발명의 일 실시예에 따른 가속화 모드를 구비한 레지스터 제어 DLL의 블럭 다이어그램.

* 도면의 주요 부분에 대한 부호의 설명

66 : 가속화 모드 단위 지연량 제어부

accel_end : 가속화 모드 종료 신호

accel-shift : 가속화 쉬프트 제어신호

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

<14> 본 발명은 반도체 회로 기술에 관한 것으로, 특히 지연고정루프(delay locked loop, DLL)에 관한 것이며, 더 자세히는 레지스터 제어 DLL(register controlled DLL)에 관한 것이다.

<15> 통상적으로, 시스템이나 회로에서 클럭은 동작 타이밍을 맞추기 위한 레퍼런스로 사용되고 있으며, 에러(error) 없이 보다 빠른 동작을 보장하기 위해서 사용되기도 한다. 외부로부터 입력되는 클럭이 내부에서 사용될 때 내부 회로에 의한 시간 지연(클럭 스큐(clock skew))이 발생하게 되는데, 이러한 시간 지연을 보상하여 내부 클럭이 외부 클럭과 동일한 위상을 갖도록 하기 위해 DLL이 사용되고 있다.

<16> 한편, DLL은 기존에 사용되어 온 위상고정루프(PLL)에 비해 잡음(noise)의 영향을 덜 받는 장점이 있어 DDR SDRAM(Double Data Rate Synchronous DRAM)을 비롯한 동기식 반도체 메모리에서 널리 사용되고 있으며, 그 중에서도 레지스터 제어 DLL이 가장 일반화되어 사용되고 있다.

<17> 동기식 반도체 메모리 소자에서 레지스터 제어 DLL은 기본적으로 외부 클럭을 받아서 클럭 경로 및 데이터 경로의 지연 성분을 보상하여 미리 네거티브 지연을 반영함으로서 데이터의 출력이 외부 클럭과 동기되도록 해주는 기능을 수행한다.

<18> 첨부된 도면 도 1은 종래기술에 따른 DDR SDRAM의 레지스터 제어 DLL의 블록 다이어그램이다. 레지스터 제어 DLL은 제1 및 제2 클럭 입력 버퍼(11, 12)로부터 출력된 내부 클럭(fclk, rclk)을 사용한다. 제1 클럭 입력 버퍼(11)는 부 외부 클럭(/CLK)을 버퍼링하여 부 외부 클럭(/CLK)의 라이징 에지(정 외부 클럭(CLK)의 폴링 에지)에 동기된 내부 클럭(fclk)을 생성하며, 제2 클럭 입력 버퍼(12)는 정 외부 클럭(CLK)을 버퍼링하여, 정 외부 클럭(CLK)의 라이징 에지에 동기된 내부 클럭(rclk)을 생성한다.

<19> 도 1을 참조하면, 종래기술에 따른 DDR SDRAM의 레지스터 제어 DLL은, 내부 클럭(rclk)을 1/M(M은 양의 정수이며, 여기서에서 M=8)로 분주하여 지연 모니터링 클럭(fb_div) 및 기준 클럭(ref)을 생성하는 클럭 분주기(13)와, 내부 클럭(fclk)을 입력으로 하는 제1 지연 라인(14)과, 내부 클럭(rclk)을 입력으로 하는 제2 지연 라인(15)과, 지연 모니터링 클럭(fb_div)을 입력으로 하는 제3 지연 라인(16)과, 제1, 제2 및 제3 지연라인(14, 15, 16)의 지연량을 결정하기 위한 쉬프트 레지스터(22)와, 지연고정시 제1 지연 라인(14)의 출력(fclk_dl)을 구동하여 DLL 클럭(fclk_dll)을 생성하기 위한 제1 DLL 드라이버(17)와, 지연고정시 제2 지연 라인(15)의 출력(rclk_dl)을 구동하여 DLL 클

력(rclk_dll)을 생성하기 위한 제2 DLL 드라이버(18)와, 제3 지연 라인(16)의 출력(fb_dl)을 입력으로 하여 실제 클럭 경로 및 데이터 경로의 지연 성분을 반영하기 위한 지연 모델(19)과, 지연 모델(19)의 출력(fb_fm)과 기준 클럭(ref)의 위상을 비교하기 위한 위상 비교기(20)와, 위상 비교기(20)의 비교 결과에 응답하여 쉬프트 레지스터(22)의 쉬프트 방향을 제어하기 위한 쉬프트 레지스터 제어기(21)를 구비한다.

<20> 이하, 상기와 같이 구성된 종래기술에 따른 레지스터 제어 DLL의 동작을 갈략히 살펴본다.

<21> 우선, 클럭 분주기(13)는 내부 클럭(rclk)을 1/M 분주하여 정 외부 클럭(CLK)의 M번째 클럭마다 한번씩 동기되는 클럭(ref, fb_div)을 만든다. 기준 클럭(ref) 및 지연 모니터링 클럭(fb_div)은 서로 반대 위상을 갖는다.

<22> 초기 동작시, 지연 모니터링 클럭(fb_div)은 제3 지연 라인(16)의 단위 지연소자 하나만을 통과한 후, 지연 모델(19)를 거치면서 예정된 지연량 만큼 되어 출력된다.

<23> 한편, 위상 비교기(20)는 기준 클럭(ref)의 라이징 에지와 지연 모델(19)의 출력 클럭(fb_dm)의 라이징 에지를 비교하고, 쉬프트 레지스터 제어기(21)는 위상 비교기(20)의 비교 결과에 응답하여 쉬프트 레지스터(22)의 쉬프트 방향을 제어하기 위한 쉬프트 제어신호(SR, SL)를 출력한다.

<24> 그리고, 쉬프트 레지스터(22)는 쉬프트 제어신호(SR, SL)에 응답하여 제1, 제2 및 제3 지연 라인(14, 15, 16)을 구성하는 다수의 단위 지연 셀 중 하나의 단위 지연 셀을 인에이블 시킴으로써 제1, 제2 및 제3 지연 라인(14, 15, 16)에 의한 지연량을 결정한다

. 이때, SR(shift right)이 인에이블 되면 쉬프트 레지스터의 값을 오른쪽으로 이동시키고, SL(shift left)이 인에이블 되면 쉬프트 레지스터의 값을 왼쪽으로 이동시킨다.

<25> 이후, 지연량이 제어된 지연 모델(19)의 출력 클럭(fb_dm)과 기준 클럭(ref)을 비교해 나가면서 두 클럭이 최소의 지터(jitter)를 가지는 순간에 지연고정(locking)이 이루어지게 되고, 이때 비로소 제1 및 제2 DLL 드라이버(17, 18)가 인에이블 되어 부 외부 클럭(/CLK) 및 정 외부 클럭(CLK)과 동일한 위상을 갖는 DLL 클럭(fclk_dll, rclk_dll)이 출력 된다.

<26> 한편, 상기와 같은 방식으로 위상고정을 이루기 위해서는 적지 않은 시간이 소요되기 때문에 레지스터 제어 DLL에 가속화 모드를 채용하게 되었다. DLL의 가속화 모드는 칩이 초기화 된 직후 외부 클럭과 내부 클럭을 동기화하는 과정에서 두 클럭의 위상차가 아주 큰 경우 지연 라인에서 증감하는 지연량을 크게 가져감으로서 빠른 시간 내에 두 클럭의 위상차가 줄어들 수 있도록 하는 DLL의 동작 모드이다.

<27> 도 2는 가속화 모드를 구비한 종래기술에 따른 DDR SDRAM의 레지스터 제어 DLL의 블록 다이어그램이다.

<28> 도 2에 도시된 DDR SDRAM의 레지스터 제어 DLL은, 제1 및 제2 입력 버퍼(31, 32), 제1, 제2 및 제3 지연 라인(34, 35, 36), 제1 및 제2 DLL 드라이버(37, 38), 지연 모델(39) 등은 상기 도 1에 도시된 일반 레지스터 제어 DLL과 동일한 구성을 가진다.

<29> 한편, 가속화 모드를 지원하기 위하여 제1 및 제2 위상 비교기(40, 44)를 구비하는데, 제1 위상 비교기(40)는 상기 도 1의 위상 비교기(20)에 상응하는 것으로 기준 클럭(ref)과 지연 모델(39)의 출력 클럭(fb_dm)을 입력으로 하며, 제2 위상 비교기(44)는 지

연 모델(39)의 출력 클럭(fb_dm)을 일정 시간(N ×unit_delay) 동안 지연시키는 지연부(43)의 출력 클럭(fbclk_dly)과 기준 클럭(ref)을 입력으로 한다. 여기서, 'N'은 2 이상의 양의 정수이며, 'unit_delay'는 지연 라인(34, 35, 36)을 이루는 단위 지연 셀의 지연량을 나타낸다. 즉, 'N ×unit_delay'는 N개의 단위 지연 셀의 지연량을 나타낸다.

<30> 그리고, 쉬프트 레지스터 제어기(41)는 제1 위상 비교기(40)의 출력(pd1)과 제2 위상 비교기(41)의 출력(ac_enz)을 입력으로 하며, 쉬프트 레지스터(42)는 쉬프트 제어신호(SR, SL)를 입력으로 한다.

<31> 도 3은 상기 도 2의 레지스터 DLL의 동작 파형도이다. 초기 동작시 기준 클럭(ref)과 지연 모델(39)의 출력 클럭(fb_dm)의 위상차(Td)가 지연부(43)에 의한 지연 시간(N × unit_delay)보다 크다면 제1 및 제2 위상 비교기(40, 44)의 출력은 모두 논리레벨 로우를 나타내고, 제2 위상 비교기(44)의 출력인 가속화 모드 인에이블 신호(ac_enz)가 인에이블 되어 쉬프트 레지스터 제어기(41)는 쉬프트 레지스터(42)가 가속화 모드를 실행하도록 한다.

<32> 참고적으로, 제1 및 제2 위상 비교기(40, 44)는 기준단과 입력단에 인가된 신호의 라이징 에지를 비교하여 입력단에 인가된 신호의 위상이 기준단에 인가된 신호의 위상보다 빠르면 논리레벨 로우를 출력하고, 기준단에 인가된 신호의 위상보다 느리면 논리레벨 하이를 출력한다.

<33> 한편, 가속화 모드를 한번 진행한 경우, 기준 클럭(ref)과 지연 모델(39)의 출력 클럭(fb_dm)의 위상차는 Td-(N ×unit_delay)로 줄어들 것이다. 이때, 위상차가 지연부(43)에 의한 지연 시간(N ×unit_delay)보다 크다면 가속화 모드 인에이블 신호(ac_enz)

가 인에이블 되어 다시 가속화 모드를 진행할 것이고, 위상차가 지연부(43)에 의한 지연 시간(N ×unit_delay)보다 작다면 가속화 모드 인에이블 신호(ac_enz)가 디스에이블 되어 가속화 모드를 멈추고 제1 위상 비교기(40)의 출력(pd1)에 의한 정상 모드(unit_delay씩 지연 제어)가 수행될 것이다.

<34> 그런데, 상기와 같은 종래기술의 문제는 제1 위상 비교기(40)의 기준단에 인가되는 기준 클럭(ref)으로 내부 클럭(rclk) 그 자체가 아니라, 그것이 1/M로 분주된 클럭을 사용한다는 것이다.

<35> 이처럼 제1 위상 비교기(40)의 입력으로 분주된 클럭을 사용하게 되면 컨트롤이 용이하고 전류 소모도 줄일 수 있는 장점이 있는 반면, 소자의 동작 속도가 높아지고 그에 따라 여러 가지 타이밍 규격(timing spec.)이 까다로와지면서 정확도 측면에서 많은 손실이 생길 수 있다.

<36> 상기와 같은 문제점을 해결하기 위하여 본 출원인은 '가속화 모드를 구비한 레지스터 제어 지연고정루프'에 관한 기술을 제안한 바 있습니다[대한민국 특허출원 제2002-66443호(2002년 10월 31일)].

<37> 도 4는 대한민국 특허출원 제2002-66443호에 따른 가속화 모드를 구비한 DDR SDRAM의 레지스터 제어 DLL의 블록 다이어그램이다.

<38> 도 4를 참조하면, 레지스터 제어 DLL은 제1 및 제2 클럭 입력 버퍼(51, 52)로부터 출력된 내부 클럭(fclk, rclk)을 사용한다. 제1 클럭 입력 버퍼(51)는 부 외부 클럭(/CLK)을 버퍼링하여 부 외부 클럭(/CLK)의 라이징 에지(정 외부 클럭(CLK)의 폴링 에지)에 동기된 내부 클럭(fclk)을 생성하며, 제2 클럭 입력 버퍼(52)는 정 외부 클럭(CLK)

을 버퍼링하여, 정 외부 클럭(CLK)의 라이징 에지에 동기된 내부 클럭(rclk)을 생성한다.

<39> 도시된 DDR SDRAM의 레지스터 제어 DLL은, 내부 클럭(fclk)을 입력으로 하는 제1 지연 라인(54)과, 내부 클럭(rclk)을 입력으로 하는 제2 지연 라인(55)과, 제1 및 제2 지연라인(54, 55)의 지연량을 결정하기 위한 쉬프트 레지스터(62)와, 지연고정시 제1 지연 라인(54)의 출력(fclk_dl)을 구동하여 DLL 클럭(fclk_dll)을 생성하기 위한 제1 DLL 드라이버(57)와, 지연고정시 제2 지연 라인(55)의 출력(rclk_dl)을 구동하여 DLL 클럭(rclk_dll)을 생성하기 위한 제2 DLL 드라이버(58)와, 제2 지연 라인(55)의 출력(rclk_dl)을 입력으로 하여 실제 클럭 경로 및 데이터 경로의 지연 성분을 반영하기 위한 지연 모델(59)과, 지연 모델(59)의 출력(fb_fm)을 N ×unit_delay만큼 지연시키기 위한 지연부(63)와, 내부 클럭(rclk)과 지연 모델(59)의 출력(fb_fm)의 위상을 비교하기 위한 제1 위상 비교기(60)와, 내부 클럭(rclk)과 지연부(63)의 출력(fbclk_dly)의 위상을 비교하기 위한 제2 위상 비교기(64)와, 제1 및 제2 위상 비교기(60, 64)의 출력(pd1, pd2)에 응답하여 가속화 모드를 계속 수행할 것인지를 결정하기 위한 모드 결정부(65)와, 제1 위상 비교기(60)의 출력(pd1)과 모드 결정부(65)의 출력에 응답하여 쉬프트 레지스터(62)의 쉬프트 모드를 제어하기 위한 쉬프트 레지스터 제어부(61)를 구비한다.

<40> 상기 도 4에 도시된 레지스터 제어 DLL은 두 개의 위상 비교기(60, 64)를 두고 있다. 제1 및 제2 위상 비교기 모두 기준단으로 내부 클럭(rclk)을 직접 입력 받으며, 제1 위상 비교기의 입력단에는 지연 모델(59)의 출력(fb_dm)이, 제2 위상 비교기의 입력단에는 지연부(63)의 출력(fbclk_dly)이 입력된다. 전술한 바와 같이 지연부(63)는 N ×

unit_delay만큼의 지연량을 가지는 바, unit_delay는 지연 라인(54, 55)을 이루는 단위 지연 셀이 가지는 지연량을 의미한다.

<41> 또한, 도시된 레지스터 제어 DLL은 DLL 클럭을 만들기 위한 클럭 소오스로서 내부 클럭(rclk) 자체를 사용하고 있다.

<42> 두 위상 비교기(60, 64) 중 제1 위상 비교기(60)는 현재 상태의 내부 클럭(rclk)과 지연 모델(59)의 출력(fb_dm)의 위상을 비교하는 역할을 수행하며, 제2 위상 비교기(64)는 현재의 지연 모델(59)의 출력(fb_dm)의 위상이 지연 라인(54, 55)에서 다시금 N × unit_delay만큼 지연되고 난 후에는 내부 클럭(rclk)과 어떤 위상 관계를 나타낼 것인가를 미리 예측하는 역할을 수행한다.

<43> 이 두 위상 비교기(60, 64)의 출력 신호(pd1, pd2)를 모드 결정부(65)가 입력 받아 가속화 모드를 제어하는데 사용한다. 만일, 현재의 지연 모델(59)의 출력(fb_dm)의 위상이 내부 클럭(rclk)보다 앞서 있고(즉, pd1가 논리레벨 로우), 현재의 지연 모델(59)의 출력(fb_dm)을 N ×unit_delay만큼 지연시킨 지연부(63)의 출력(fbclk_dly)의 위상 역시 내부 클럭(rclk)보다 앞서 있는 경우(즉, pd2가 논리레벨 로우)라면, 지연 라인(54, 55)에서 N개의 단위 지연 셀만큼 지연을 더 가하더라도 지연 모델(59)의 출력(fb_dm)은 내부 클럭(rclk)보다 위상이 앞서있을 것임을 의미하는 것이다. 곧, 제1 및 제2 위상 비교기(60, 54)의 출력(pd1, pd2)이 모두 논리레벨 로우인 경우에는 모드 결정부(65)의 출력인 가속화 모드 종료 신호(accel_end)가 논리레벨 로우가 되어 계속해서 가속화 모드를 동작시키라는 명령을 쉬프트 레지스터 제어부(61)에 인가하게 되며, 쉬프트 레지스터 제어부(61)에서는 가속화 쉬프트 제어신호(accel-shift)를 인에이블 시켜서 실제로 지

연 라인(54, 55)에 입력되는 내부 클럭(rclk)을 N개의 단위 지연 셀의 지연량만큼 더 지연시켜서 출력하게 된다.

<44> 도 5는 상기 도 4의 레지스터 제어 DLL의 동작 파형을 예시한 도면이다.

<45> 레지스터 제어 DLL에서 지연고정이라 함은 지연 모델(59)의 출력(fb_dm)의 화살표로 표시된 라이징 에지가 클럭 소오스인 내부 클럭(rclk)의 화살표로 표시된 라이징 에지와 일치됨을 의미하는 것으로, 레지스터 제어 DLL에서는 비교되는 두 클럭의 라이징 에지를 근접시키는 동작을 진행하게 된다.

<46> 도 5를 참조하면, 초기 동작시 지연 모델(59)의 출력(fb_dm)과 지연부(63)의 출력(fbclk_dly)의 라이징 에지의 위상이 내부 클럭(rclk)의 라이징 에지보다 앞서 있기 때문에 가속화 모드가 인에이블 된다(accel_end가 논리레벨 로우). 가속화 모드를 한번 수행하게 되면 지연 라인(54, 55)에서 내부 클럭(rclk)을 N ×unit_delay만큼 더 지연을 시켜서 출력을 하게 되는데, 이렇게 되면 1번의 가속화 모드 동작후의 지연 모델(59)의 출력(fb_dm)의 라이징 에지의 위상은 가속화 모드 동작을 하기 전의 지연부(63)의 출력(fbclk_dly)의 라이징 에지와 동일한 위상을 갖게 된다. 도면의 예에서는 1번의 가속화 모드 동작 후에도 역시 지연 모델(59)의 출력(fb_dm)과 지연부(63)의 출력(fbclk_dly)의 라이징 에지의 위상이 모두 내부 클럭(rclk)의 라이징 에지의 위상보다 앞서 있기 때문에 계속해서 가속화 모드를 수행하게 되는데, 이런 방식으로 3번의 가속화 모드를 수행한 후에 지연 모델(59)의 출력(fb_dm) 및 지연부(63)의 출력(fbclk_dly)의 위상과 내부 클럭(rclk)의 위상을 비교해 보면 지연 모델(59)의 출력(fb_dm)은 여전히 내부 클럭(rclk)의 위상보다 앞서 있지만, 앞으로 1번 더 가속화 모드를 동작시킨 후의 지연 모델(59)의 출력(fb_dm)의 위상을 예측한 지연부(63)의 출력(fbclk_dly)의 위상은 내부 클럭

(rclk)의 위상보다 뒤쳐지게 되므로, 모든 가속화 모드는 여기서 끝내야 한다(accel_end가 논리레벨 로우).

<47> 한편, 각 가속화 모드 동작 사이에는 지켜야 할 시간 간격이 있는데, 그 값은 내부 클럭(rclk)이 지연 라인(54, 55)을 통과하는 시간, 그 출력이 다시 지연 모델(59)을 거치는 시간, 또 그 출력이 지연부(63)와 제2 위상 비교기(64)를 거치는 모든 시간의 합(tTA)보다 커야한다. 그 이유는 1번의 가속화 모드가 진행되면 지연 라인(54, 55)에서 지연 동작이 일어나서 그 출력이 지연 모델(59), 지연부(63), 제2 위상 비교기(64)를 거쳐서 제1 및 제2 위상 비교기(60, 64)의 출력(pd1, pd2)의 값을 새롭게 업데이트 한 후에 다시금 가속화 모드를 계속해야 할 지 아니면 중지해야 할 지를 결정해야 하기 때문이다. 이 시간을 지키지 않으면 이전의 가속화 모드에 의해서 새롭게 위상이 바뀐 신호를 가지고 제1 및 제2 위상 비교기(60, 64)에서 비교되어 pd1과 pd2가 만들어지지 못하기 때문에 오동작이 발생할 수가 있다.

<48> 도 6은 상기 도 5의 쉬프트 레지스터(62) 및 제2 지연 라인(55)의 회로 구성을 예시한 도면이다.

<49> 도 6을 참조하면, 쉬프트 레지스터(62)는 각각 리셋(reset)단(표시되지 않음)과, 정출력단(Q) 및 부출력단(Qb)을 가지는 다수의 래치(…, Ln~Ln+7,…)를 구비한다. 한편, 각 래치의 래치값을 제어하기 위해 즉, 래치간에 단위 쉬프트 동작을 유도하기 위하여 이웃한 래치는 스위치를 통해 연결되도록 되어 있으며, 이 스위치를 제어하는 신호가 쉬프트 레지스터 제어부(61)로부터 출력된 단위 쉬프트 제어신호(SR, SL)이다. 그리고, N(여기에서는 '3'으로 가정함)개 만큼 떨어진 래치의 출력값이 스위치를 통해 연결될 수

있도록 묶여 있으며, 이 스위치를 제어하는 신호가 쉬프트 레지스터 제어부(61)로부터 출력된 가속화 쉬프트 제어신호(accel-shift)이다.

<50> 또한, 쉬프트 레지스터(62)는 하나의 단위 지연 셀(UDC)를 선택하기 위하여 각 래치에 대응하는 다수의 노아 게이트를 구비한다. 예컨대, n번째 노아 게이트(NOR1)는 n-1번째 래치(도시하지 않음)의 부출력(Qb)과 n+1번째 레지스터의 정출력(Q)을 입력으로 한다.

<51> 한편, 제2 지연 라인(55)은 상기 노아 게이트(NOR)의 출력을 일입력으로 하고, 내부 클럭(rclk)을 타입력으로 하는 다수의 낸드 게이트(NAND1)와, 낸드 게이트(NAND1) 각각의 출력을 전파하는 다수의 단위 지연 셀(UDC)을 구비한다. 각 단위 지연 셀(UDC)은 낸드 게이트(NAND1)의 출력을 일입력으로 하고, 전단 UDC의 출력을 타입력으로 하는 낸드 게이트(NAND2)와, 공급전원(VDD)를 일입력으로 하고, 낸드 게이트(NAND2)의 출력을 타입력으로 하는 낸드 게이트(NAND)를 구비한다.

<52> 참고적으로, 제1 지연 라인(54)의 구성은 입력 클럭을 제외하고는 전술한 제2 지연 라인(55)의 구성과 동일하다.

<53> 이하, 상기 도 6에 도시된 회로의 동작을 간략히 살펴본다.

<54> 칩이 초기화되면 쉬프트 레지스터(62)의 각 래치가 초기화된다.

<55> 이어서, DLL이 초기 동작을 수행하고, 제1 및 제2 위상 비교기(60, 64)의 출력(pd1, pd2)에 따라 모드 결정부(65)는 가속화 모드 종료 신호(accel_end)를 인에이블/디스에이블 시켜 가속화 모드를 진행할 것인지 가속화 모드를 중단하고 정상 모드를 수행할 것인지를 결정한다. 만일 가속화 모드 종료 신호(accel_end)가 인에이블 되어 가속화

모드가 중단되면 제1 위상 비교기(60)의 출력(pd1)에 응답하여 쉬프트 레지스터 제어부(61)에서 단위 쉬프트 제어신호(SR, SL)를 출력하여 정상 모드를 진행한다.

<56> 쉬프트 레지스터(62)는 다수의 노아 게이트 중 어느 하나의 출력만을 논리레벨 하이로 만들어 그 출력을 입력 받은 낸드 게이트를 통해 내부 클럭(rclk)을 통과시키고, 해당 낸드 게이트의 출력단에 연결된 단위 지연 셀(UDC)을 선택하게 된다. 이처럼 단위 지연 셀(UDC)이 선택되면, 내부 클럭(rclk)이 거치는 단위 지연 셀(UDC)의 수가 결정된다.

<57> 쉬프트 레지스터 제어부(61)의 출력 중 SR(shift right)은 래치의 출력값을 이웃하는 오른쪽 래치에 전달하라는 제어신호이며, SL(shift left)은 래치의 출력값을 이웃하는 왼쪽 래치에 전달하라는 제어신호이며, 가속화 쉬프트 제어신호(accel-shift)는 래치의 출력값을 왼쪽으로 N개만큼 떨어져있는 래치에 전달하라는 제어신호이다. 예컨대, 현재 s개 만큼의 단위 지연 셀(UDC)을 거쳐 내부 클럭(rclk)이 출력되고 있는 상황을 가정하면, SR 신호가 한번 활성화되면 내부 클럭(rclk)이 거치는 단위 지연 셀(UDC)의 수는 s-1개가 될 것이고, SL 신호가 한번 활성화된다면 내부 클럭(rclk)이 거치는 단위 지연 셀(UDC)의 수는 s+1개가 될 것이며, 가속화 쉬프트 제어신호(accel-shift)가 한 번 활성화된다면 내부 클럭(rclk)이 거치는 단위 지연 셀(UDC)의 수는 s+N개가 될 것이다.

<58> 하기의 표 1은 모드 결정부(65)의 동작 진리표이다.

<59> 【표 1】

현재 accel_end	pd1	pd2	이전 pd2	다음 accel_end
1	don't care			1
0	0	1	don't care	1
0	don't care	1	0	1
0	anything else(& reset value)			0

<60> 상기 표 1을 참조하면, 가속화 모드 종료 신호(accel_end)의 초기값은 '0'인데, 가속화 모드 종료 신호(accel_end)가 '0'일때는 가속화 모드를 수행할 수 있는 상황을 의미하고, 가속화 모드 종료 신호(accel_end)가 '1'일때는 내부 클럭(rclk)과 지연 모델(59)의 출력(fb_dm)이 어느 정도 가까워져서 가속화 모드가 종료됨을 의미한다. 가속화 모드 종료 신호(accel_end)의 초기값이 '0'이므로 초기에는 항상 가속화 모드를 수행할 준비가 되어 있다.

<61> 먼저, 현재 가속화 모드 종료 신호(accel_end)가 '1'일때는 pd1, pd2, 이전 pd2 값에 관계 없이 다음 가속화 모드 종료 신호(accel_end)는 '1'이 된다.

<62> 다음으로, 현재 가속화 모드 종료 신호(accel_end)가 '0'일때는 pd1, pd2, 이전 pd2 값에 따라 여러 가지 경우가 나타나게 된다.

<63> 도 7a는 pd1이 '0'이고 pd2가 '1'인 경우의 파형도로서, 가속화 모드가 끝나는 시점에서 주로 나타나는 파형을 나타내고 있다. 이 경우, 이전 pd2의 값에 관계 없이 다음 가속화 모드 종료 신호(accel_end)는 '1'이며, 이는 가속화 모드가 종료됨을 의미한다.

<64> 도 7b는 pd2가 '1'이고 이전 pd2가 '0'인 경우의 파형도로서, k번째 가속화 모드 동작 후의 파형과 k+1번째 가속화 모드 동작 후의 파형을 나타내고 있다. k번째 가속화 모드 동작 후의 상태가 pd1 및 pd2가 모두 '0'을 나타내고 있으나, 지연부(63)의 출력(fbclk_dly)의 라이징 에지와 내부 클럭(rclk)의 라이징 에지의 위상차가 충분히 작은 값을 가진다. 이 경우 이상적으로는 k+1번째 가속화 모드 동작 후의 지연 모델(59)의 출력(fb_dm)의 위상은 k번째 가속화 모드 동작 후의 지연부(63)의 출력(fbclk_dly)의 위상과 같아야 한다. 그러나, 제2 위상 비교기(64)의 전단에 배치된 지연부(63)의 지연량(N×unit_delay)과 지연 라인(54, 55)에서의 N개의 단위 지연 셀(UDC)이 가지는 지연량에는

미세하나마 차이가 발생할 수 있다. 그 이유는 각각의 입출력 조건(입력 슬로프, 출력 로딩)이 다르기 때문이다. 따라서, k+1번째 가속화 모드 동작 후에 지연부(63)의 출력(fbclk_dly)은 물론 지연 모델(59)의 출력(fb_dm)이 내부 클럭(rclk)의 위상보다 뒤쳐지는 경우가 발생할 수 있으며, 이러한 경우에 대해서도 가속화 모드가 종료됨을 제대로 결정해 줄 수 있어야 한다(상기 도 7a의 경우만 가지고는 도 7b의 경우를 대처할 수 없음). 그러므로, k번째 가속화 모드 동작 후의 pd2(이전 pd2)가 '0'이고, k+1번째 가속화 모드 동작 후의 pd2가 '1'인 경우에도 가속화 모드 종료 신호(accel_end)는 '1'로 출력되어 가속화 모드가 종료된다.

<65> 그리고, 위에서 언급하지 않은 모든 경우에는 가속화 모드 종료 신호(accel_end)가 '0'이 되어 가속화 모드 동작을 수행하면 되며, 초기화 시에도 이에 해당한다.

<66> 한편, 상기와 같이 pd2의 이전 상태를 파악하기 위해서는 모드 결정부(65)의 pd2 신호 입력단에 래치를 구비해야 하며, 가속화 모드의 중단 상태를 계속 유지하기 위해서 모드 결정부(65)의 가속화 모드 종료 신호(accel_end) 출력단에 래치를 구비해야 한다.

<67> 도 8은 상기 도 4의 지연부(63)의 지연량(N ×unit_delay)을 결정하는 'N'값의 범위를 설명하기 위한 도면이다.

<68> 전술한 종래기술에서는 분주된 클럭이 아닌 프리 러닝(free-running)하는 클럭을 사용하여 가속화 동작 모드를 구현하고 있기 때문에 1 tCK 마다 계속해서 라이징 에지가 나타나는 악조건 속에서 가속화 모드를 언제 중지할 것인지를 제대로 파악해야 한다. 지연부(63)의 지연량(N ×unit_delay)이 타겟으로 하는 최대 동작

주파수(tCK,min)의 반주기에 해당하는 값보다 크게 되면 실제는 가속화 모드 동작을 해서는 안되는데도 불구하고 가속화 모드 종료 신호(accel_end)가 '0'이 되어 고속화 모드를 수행하는 경우가 발생할 수 있다. 다시 말해, 지연 모델(59)의 출력(fb_dm)의 라이징 에지가 내부 클럭(rclk)의 논리레벨 하이 구간 내에 존재하고, 지연부(63)의 출력(fbclk_dly)의 라이징 에지가 내부 클럭(rclk)의 논리레벨 로우 구간을 넘어서서 존재하는 경우가 발생하면 오동작을 일으킬 우려가 있다. 그 이유는 지연 모델(59)의 출력(fb_dm)과 지연부(63)의 출력(fbclk_dly) 모두의 라이징 에지가 내부 클럭(rclk)의 연속된 논리레벨 하이 구간 내에 있는지 아니면 두 신호의 라이징 에지가 각각 내부 클럭(rclk)의 다른 논리레벨 하이 구간 내에 있는지를 위상 비교기가 파악할 수 없기 때문이다. 따라서, 허용되는 'N'값의 범위는 지연부(63)의 지연량(N ×unit_delay)이 타겟으로 하는 최대 동작 주파수의 반주기(1/2×tCK,min)보다 작은 범위 내에서 결정하여야 한다.

<69> 예컨대, 타겟으로 하는 최대 동작 주파수가 333MHz이고(tCK,min=3ns), 단위 지연셀(UDC)의 지연값이 150ps라고 가정하면, 하기의 수학식 1이 성립된다.

<70> 【수학식 1】 1/2×3ns > N ×150ps

<71> 따라서, N < 10이라는 결론이 나온다.

<72> 그런데, 전술한 종래기술에서는 가속화 모드에서 한 번에 지연량을 조절할 수 있는 양이 동작 주파수에 관계 없이 'N ×unit_delay'로 고정되며, N값은 최대 동작 주파수에 의해 특정값으로 제한될 수 밖에 없었다.

<73> 통상적으로, 동작 주파수가 낮으면 DLL 블럭에서 지연시켜야 하는 지연량이 많아지며, 그에 따라 지연고정 시간 또한 늘어난다. 전술한 종래기술의 경우, 최대 동작 주파수에 의해 정해진 N값이 동작 주파수에 관계 없이 고정될 수 밖에 없기 때문에 고주파 동작시와 저주파 동작시의 지연고정 시간이 달라지게 된다.

<74> 즉, 시스템이 저주파 동작하는 경우, 고주파 동작시에 비해 DLL 블럭에 의한 지연량이 늘어남에도 불구하고, 작은 N값으로 가속화 모드를 수행하기 때문에 고주파 동작시에 비해 지연고정 시간이 증가하는 문제점이 있었다.

【발명이 이루고자 하는 기술적 과제】

<75> 본 발명은 상기와 같은 종래기술의 문제점을 해결하기 위하여 제안된 것으로, 동작 주파수에 따른 지연고정 시간의 차이를 개선할 수 있는 가속화 모드를 구비한 레지스터 제어 지연고정루프를 제공하는데 그 목적이 있다.

【발명의 구성 및 작용】

<76> 상기의 기술적 과제를 달성하기 위한 본 발명의 일 측면에 따르면, 가속화 모드를 구비한 레지스터 제어 지연고정루프에 있어서, 내부 클럭을 지연시키기 위한 다수의 단위 지연 셀을 구비하는 지연 라인; 상기 지연 라인을 통과한 상기 내부 클럭에 실제 클럭 경로의 지연 조건을 반영하기 위한 지연 모델; 상기 지연 모델의 출력 신호를 일정 시간만큼 지연시키기 위한 지연 수단; 상기 지연 모델의 출력 신호와 상기 내부 클럭의 위상을 비교하기 위한 제1 위상 비교 수단; 상기 지연 수단의 출력 신호와 상기 내부 클

럭의 위상을 비교하기 위한 제2 위상 비교 수단; 상기 제1 및 제2 위상 비교 수단의 출력 신호에 응답하여 가속화 모드의 진행/중단 여부를 결정하기 위한 모드 결정 수단; 상기 제1 위상 비교 수단 및 상기 모드 결정 수단의 출력 신호에 응답하여 쉬프트 레프트 신호, 쉬프트 라이트 신호, 가속화 쉬프트 신호를 출력하기 위한 쉬프트 레지스터 제어 수단; 동작 주파수 정보에 따라 가속화 모드에서의 단위 지연량을 조절하기 위한 가속화 모드 단위 지연량 제어 수단; 및 상기 쉬프트 레지스터 제어 수단의 출력 및 상기 가속화 모드 단위 지연량 제어 수단에 응답하여 상기 지연 라인의 지연량을 제어하기 위한 쉬프트 레지스터를 구비하는 레지스터 제어 지연고정루프가 제공된다.

<77> 본 발명은 DLL의 가속화 모드의 단위 지연량 'N ×tnit_delay'의 N값을 최대 동작 주파수에 맞춰 고정시켜 사용하지 않고 동작 주파수에 따라 유동적으로 사용한다. 즉, 저주파 동작시에는 N값을 키워 가속화 모드의 단위 지연량을 증가시키고, 고주파 동작시에는 N값을 줄여 가속화 모드의 단위 지연량을 줄인다. 이 경우, 저주파 동작이든 고주파 동작이든 가속화 모드로 동작하는 사이클은 거의 비슷하지만 가속화 모드의 단위 지연량이 서로 달라져서, 결국 전체 사이클 측면에서는 지연고정 시간이 비슷해진다. 문제는 동작 주파수를 어떻게 파악하여 그에 대응하는 N값을 결정하는가 인데, 예컨대 동기식 반도체 메모리 소자의 경우에는 동작 주파수와 밀접한 관계가 있는 카스 레이턴시(CAS Latency), 모드 레지스터 세팅 등의 인덱스를 이용하면 된다.

<78> 이하, 본 발명이 속한 기술분야에서 통상의 지식을 가진 자가 본 발명을 보다 용이하게 실시할 수 있도록 하기 위하여 본 발명의 바람직한 실시예를 소개하기로 한다.

<79> 도 9는 본 발명의 일 실시예에 따른 가속화 모드를 구비한 레지스터 제어 DLL의 블럭 다이어그램이다.

<80> 도 9를 참조하면, 본 실시예에 따른 가속화 모드를 구비한 레지스터 제어 DLL은 상기 도 4에 도시된 종래기술과 거의 유사한 구성을 가진다. 따라서, 도 5에서 상기 도 4와 동일한 구성부에 대해서는 동일한 도면 부호를 사용하였다.

<81> 다만, 본 실시예에 따른 가속화 모드를 구비한 레지스터 제어 DLL은 상기 도 4의 구성을 모두 포함하며, 동작 주파수 정보 신호(freq_info)에 응답하여 가속화 모드에서 쉬프트 레지스터(62)의 단위 쉬프트량을 제어하기 위한 가속화 모드 단위 지연량 제어부(66)를 더 구비한다.

<82> 가속화 모드 단위 지연량 제어부(66)는 가속화 모드에서 쉬프트 레지스터(62)의 단위 쉬프트량 'N ×unit_delay'의 N값을 조절하는 역할을 한다. 즉, 본 실시예에 따르면 시스템의 동작 주파수에 따라 N값을 조절할 수 있다.

<83> 이하, 본 실시예에 따른 가속화 모드를 구비한 레지스터 제어 DLL의 동작을 살펴본다.

<84> 본 실시예에 따른 가속화 모드를 구비한 레지스터 제어 DLL의 동작은 동작 주파수에 따라 N값을 조절하는 것을 제외하고는 상기 도 4의 종래기술에 따른 레지스터 제어 DLL의 동작과 동일하다. 따라서, 이하에서는 가속화 모드 단위 지연량 제어부(66)의 동작을 중심으로 설명하기로 한다.

<85> 전술한 수학식 1과 같이 최대 동작 주파수을 기준으로 얻어진 N값으로 가속화 모드를 수행하는 경우, 낮은 동작 주파수에서 시스템이 동작한다면 지연 고정 시간이 늘어나

게 됨을 앞에서 언급한 바 있다. 한편, 본 실시예에서는 동작 주파수 정보 신호(freq_info)를 이용하여 N값을 조절함으로써 가속화 모드에서의 쉬프트 레지스터(62)의 쉬프트량을 제어할 수 있다. 즉, 낮은 동작 주파수에서는 N값을 증가시키고, 높은 동작 주파수에서는 N값을 낮춤으로써 동작 주파수가 바뀌더라도 지연 고정 시간을 일정 수준으로 확보할 수 있다.

<86> 이처럼 동작 주파수에 따라 다른 N값을 선택하여 사용하는 경우, 상기 수학식 1은 하기의 수학식 2와 같이 바뀌어야 할 것이다.

<87> 【수학식 2】 1/2 ×tCK,current > N ×tnit_delay

<88> 따라서, N값은 현재의 동작 주파수에 의해 가변적이다. 이처럼 가변적인 N값을 결정함에 있어서, 현재의 동작 주파수의 한 주기(tCK,current)의 1/2에 해당하는 값보다 'N ×tnit_delay' 값이 작은 것을 만족하는 N값이면 어떤 값이든 허용이 된다.

<89> 동작 주파수가 낮아짐에 따라 tCK,current 값은 점점 커지므로 자연히 가속화 모드로 동작할 때의 N값은 조금씩 커지게 된다. 따라서, N값이 커지게 되면 가속화 모드에서의 단위 지연량이 커져서 낮은 동작 주파수에서 지연 고정 시간이 빨라지게 된다.

<90> 한편, 현재의 동작 주파수를 파악하기 위한 동작 주파수 정보 신호(freq_info)는 현재의 동작 주파수와 밀접한 관련을 가지는 인덱스를 사용하면 되며, 가속화 모드 단위 지연량 제어부(66)는 동작 주파수 정보 신호(freq_info)에 대응하는 N값의 정보를 가지고 있어야 한다.

<91> 동기식 반도체 메모리 소자의 경우라면 동작 주파수 정보 신호(freq_info)를 생성하기 위하여 카스 레이턴시(CAS latency, CL)나 모드 레지스터 세팅값을 이용하면 된다.

<92> 우선, 카스 레이턴시를 이용하는 경우를 설명한다.

<93> 통상적으로, 반도체 메모리 소자의 동작 주파수가 높은 경우에는 카스 레이턴시를 큰 값으로 설정하며, 동작 주파수가 낮은 경우에는 카스 레이턴시를 작은 값으로 설정한다.

<94> 카스 레이턴시(CL)가 2, 3, 4가 존재하고 각 CL값에 대응하는 동작 주파수 범위가 아래와 같은 경우를 가정한다.

<95> CL = 2 : 100MHz~200MHz

<96> CL = 3 : 200MHz~300MHz

<97> CL = 4 : 300MHz~400MHz

<98> 예컨대, CL이 2인 경우에는 최대 동작 주파수가 200MHz(tCK,current=5ns)이므로, 이를 상기 수학식 2에 적용하면 N < 12.5가 된다. 같은 방식으로 CL이 3일 경우에는 N < 8.25, CL이 4일 경우에는 N < 6.25가 된다. 이때, 각 주파수 범위별 최대 동작 주파수에 대한 조건(수학식 1 참조)을 만족하면서 N값이 최대의 값(지연 고정 시간을 최소화하기 위한 값)을 가져야 하기 때문에, CL이 2인 경우에는 N값을 12로 세팅하고, CL이 3인 경우에는 N값을 8, CL이 4인 경우에는 N값을 6으로 세팅하며, 현재의 동작 주파수에 따라 N값을 가변적으로 채택하여 가속화 모드를 수행할 수 있다.

<99> 다음으로, 모드 레지스터 세팅값을 이용하는 경우를 설명한다.

<100> 통상적으로, 반도체 메모리 소자에는 메모리 동작의 전반적인 모드를 결정하는 모드 레지스터가 존재하는데, 이 레지스터에서 현재 사용되지 않는 특정 비트를 이용하여 이를 가속화 모드에서 N값을 결정하는 정보로 사용할 수 있다.

<101> 모드 레지스터에서 A9번과 A10번에 해당하는 두 비트를 N값을 결정하기 위한 용도로 사용하고, A9 및 A10에 대한 디코딩 값이 아래와 같다고 가정한다.

<102> A10, A9 = '00' : N = 12

<103> A10, A9 = '01' : N = 8

<104> A10, A9 = '10' : N = 6

<105> 반도체 메모리 소자의 초기화 과정에서 미리 MRS(mode register set) 명령으로 A10 및 A9에 위의 경우 중 어느 한 가지를 세팅한 후 동작을 시키게 되면 DLL의 가속화 모드에서 A10, A9의 2비트 값에 대응하는 N값으로 동작할 수 있다. 예컨대, 가장 낮은 동작 주파수 대역에서는 A10, A9를 '00'으로 세팅하고, 가장 높은 동작 주파수 대역에서는 A10, A9를 '10'으로 세팅하면 N값을 가변적으로 채택할 수 있다.

<106> 한편, 위에서는 N값을 12, 8, 6의 세가지 경우로 가변하는 경우를 가정하였는데, 이처럼 N값을 가변시키기 위해서는 쉬프트 레지스터(62)에 이를 반영하기 위한 설계적 고려가 필요하다.

<107> 이상에서 설명한 바와 같이 본 발명을 적용하면 가속화 모드에서의 단위 지연량을 유동적으로 조절할 수 있기 때문에 낮은 동작 주파수에서 지연 고정 시간이 증가하는 현상을 방지할 수 있다.

<108> 이상에서 설명한 본 발명은 전술한 실시예 및 첨부된 도면에 의해 한정되는 것이 아니고, 본 발명의 기술적 사상을 벗어나지 않는 범위 내에서 여러 가지 치환, 변형 및 변경이 가능하다는 것이 본 발명이 속한 기술분야에서 통상의 지식을 가진 자에게 있어 명백할 것이다.

<109> 예컨대, 전술한 실시예에서는 외부 클럭(CLK)의 라이징 에지에 동기된 내부 클럭(rclk)을 클럭 소오스로 사용하는 경우를 일례로 들어 설명하였으나, 외부 클럭(CLK)의 폴링 에지에 동기된 내부 클럭(fclk)을 클럭 소오스로 사용하는 경우에도 적용된다.

<110> 또한, 전술한 실시예에서는 DDR SDRAM의 레지스터 제어 DLL을 일례로 들어 설명하였으나, 본 발명의 레지스터 제어 DLL은 다른 동기식 반도체 메모리나 기타 동기식 로직에도 적용할 수 있다.

<111> 또한, 전술한 실시예에서는 지연부의 지연량과 가속화 모드에 의해 점프하는 지연 라인의 지연량이 동일한 경우를 일례로 하여 설명하였으나, 가속화 모드의 기술적 원리상 가속화 모드에 의해 점프하는 지연 라인의 지연량이 지연부의 지연량을 초과하지 않으면 된다.

【발명의 효과】

<112> 전술한 본 발명은 낮은 동작 주파수에서 DLL의 지연 고정 시간이 증가하는 현상을 방지할 수 있으며, 이로 인하여 지연 고정 시간을 거의 일정하게 유지하여 DLL의 동작 특성을 개선하는 효과가 있다.

【특허청구범위】

【청구항 1】

가속화 모드를 구비한 레지스터 제어 지연고정루프에 있어서,

내부 클럭을 지연시키기 위한 다수의 단위 지연 셀을 구비하는 지연 라인;

상기 지연 라인을 통과한 상기 내부 클럭에 실제 클럭 경로의 지연 조건을 반영하기 위한 지연 모델;

상기 지연 모델의 출력 신호를 일정 시간만큼 지연시키기 위한 지연 수단;

상기 지연 모델의 출력 신호와 상기 내부 클럭의 위상을 비교하기 위한 제1 위상 비교 수단;

상기 지연 수단의 출력 신호와 상기 내부 클럭의 위상을 비교하기 위한 제2 위상 비교 수단;

상기 제1 및 제2 위상 비교 수단의 출력 신호에 응답하여 가속화 모드의 진행/중단 여부를 결정하기 위한 모드 결정 수단;

상기 제1 위상 비교 수단 및 상기 모드 결정 수단의 출력 신호에 응답하여 쉬프트 레프트 신호, 쉬프트 라이트 신호, 가속화 쉬프트 신호를 출력하기 위한 쉬프트 레지스터 제어 수단;

동작 주파수 정보에 따라 가속화 모드에서의 단위 지연량을 조절하기 위한 가속화 모드 단위 지연량 제어 수단; 및

상기 쉬프트 레지스터 제어 수단의 출력 및 상기 가속화 모드 단위 지연량 제어 수단에 응답하여 상기 지연 라인의 지연량을 제어하기 위한 쉬프트 레지스터

를 구비하는 레지스터 제어 지연고정루프.

【청구항 2】

제1항에 있어서,

상기 동작 주파수 정보는 카스 레이턴시인 것을 특징으로 하는 레지스터 제어 지연고정루프.

【청구항 3】

제1항에 있어서,

상기 동작 주파수 정보는 모드 레지스터의 특정 비트 값인 것을 특징으로 하는 레지스터 제어 지연고정루프.

【청구항 4】

제1항에 있어서,

상기 지연 수단의 지연량은 상기 가속화 쉬프트 신호에 응답하여 증가하는 상기 지연 라인의 지연량과 실질적으로 동일한 것을 특징으로 하는 레지스터 제어 지연고정루프.

【청구항 5】

제1항에 있어서,

상기 모드 결정 수단은,

상기 제2 위상 비교 수단의 출력 신호를 래치하기 위한 제1 래치를 구비하는 것을 특징으로 하는 레지스터 제어 지연고정루프.

【청구항 6】

제5항에 있어서,

상기 모드 결정 수단은,

자신의 출력 신호를 래치하기 위한 제2 래치를 구비하는 것을 특징으로 하는 레지스터 제어 지연고정루프.

【청구항 7】

제2항에 있어서,

상기 지연 수단의 지연량은 상기 단위 지연 셀의 지연량의 정수배에 해당하는 것을 특징으로 하는 레지스터 제어 지연고정루프.

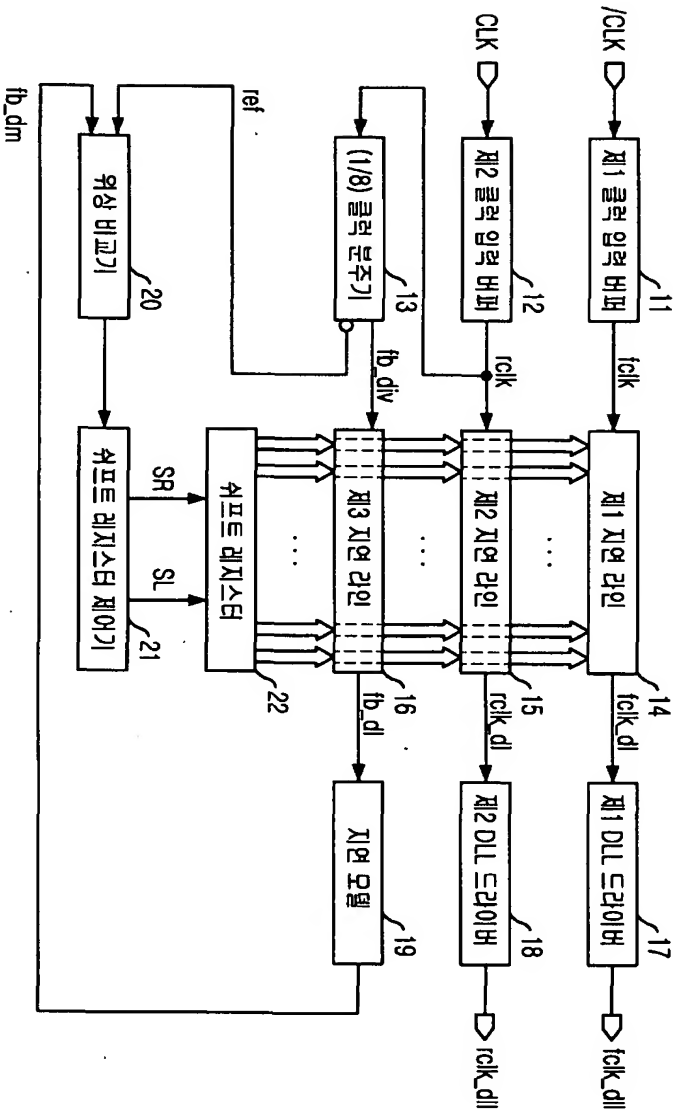
【청구항 8】

제1항 또는 제7항에 있어서,

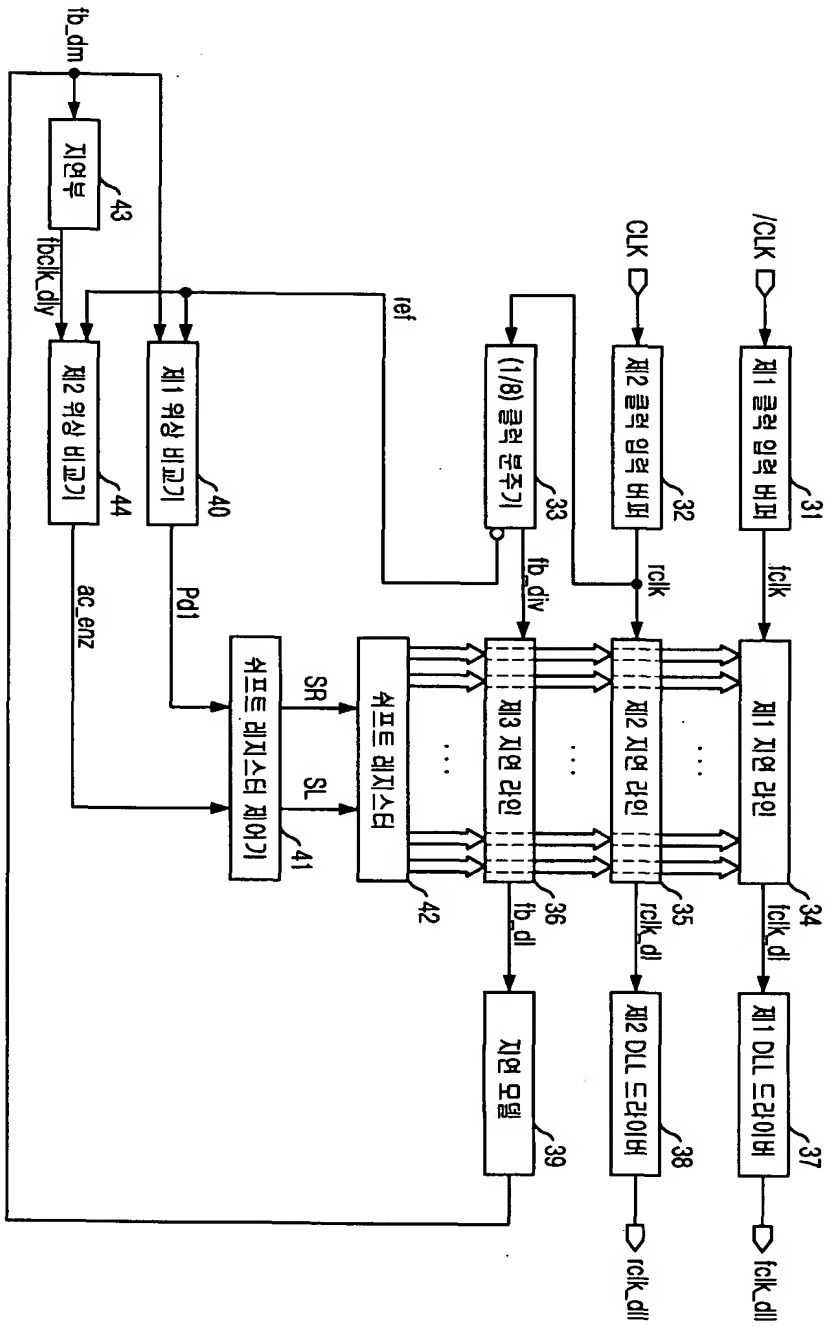
상기 지연 수단의 지연량은 상기 내부 클럭의 주파수의 반주기보다 작은 것을 특징으로 하는 레지스터 제어 지연고정루프.

【도면】

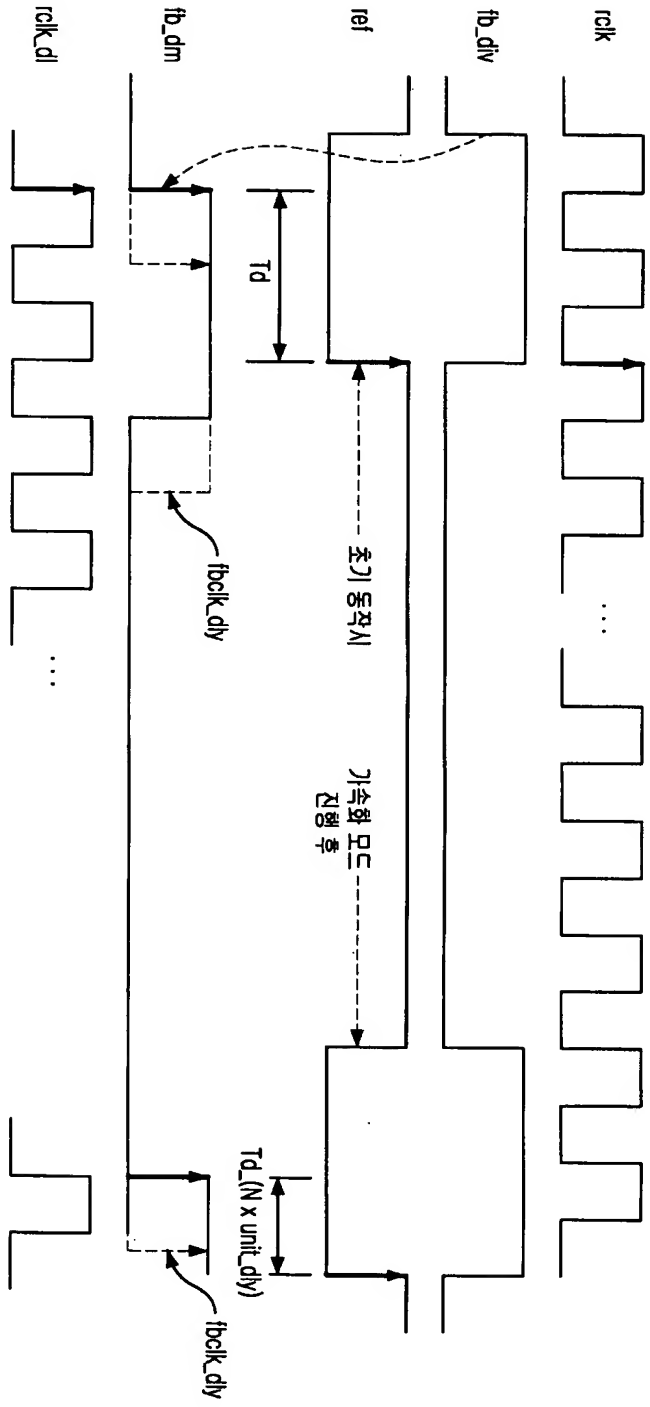
【도 1】


/CLK
CLK
fb_dm
ref
제1 클럭 입력 버퍼
제2 클럭 입력 버퍼
(1/8) 클럭 분주기
위상 비교기
fclk
rclk
fb_div
제1 지연 라인
제2 지연 라인
제3 지연 라인
쉬프트 레지스터
쉬프트 레지스터 제어기
SR
SL
fclk_dl
rclk_dl
fb_dl
제1 DLL 드라이버
제2 DLL 드라이버
지연 모델
fclk_dll
rclk_dll
11
12
13
14
15
16
17
18
19
20
21
22

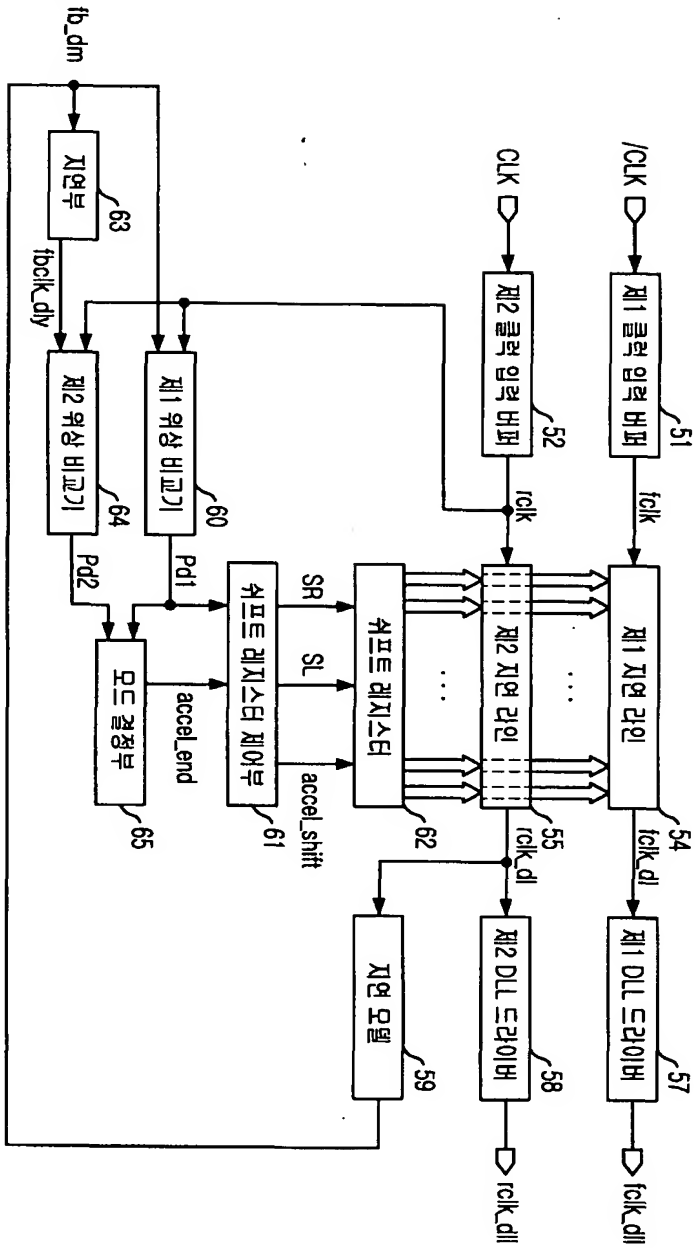
【도 2】


fb_dm
지연부
43
fbclk_dly
제2 위상 비교기
44
제1 위상 비교기
40
ref
(1/8) 클럭 분주기
33
CLK
제2 클럭 입력 버퍼
32
/CLK
제1 클럭 입력 버퍼
31
fb_div
rclk
fclk
ac_enz
Pd1
쉬프트 레지스터 제어기
41
SR
SL
쉬프트 레지스터
42
제3 지연 라인
제2 지연 라인
제1 지연 라인
36
35
34
fb_dl
rclk_dl
fclk_dl
지연 모델
39
제2 DLL 드라이버
38
제1 DLL 드라이버
37
rclk_dll
fclk_dll

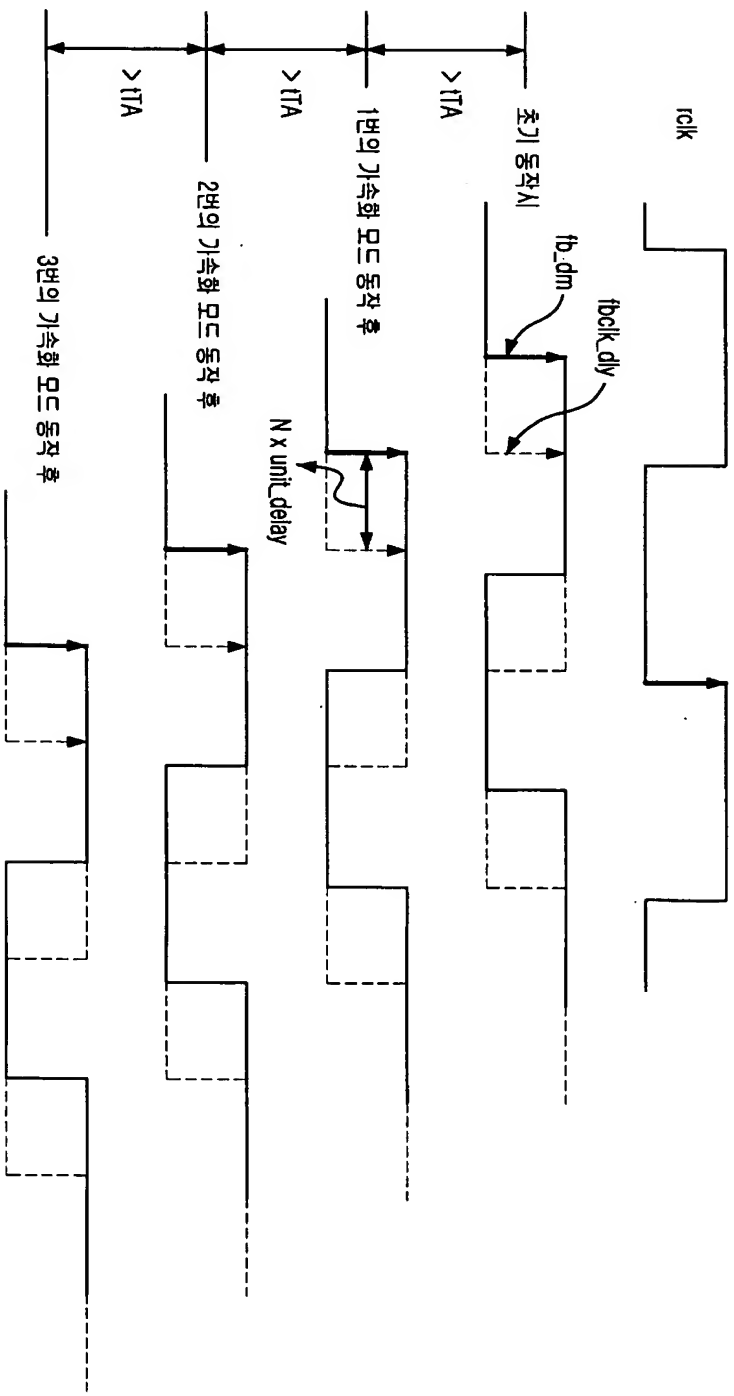
【도 3】


rclk_dl
fb_dm
ref
fb_div
rclk
Td
초기 동작시
가속화 모드
진행 후
fbclk_dly
Td_(N x unit_dly)
fbclk_dly

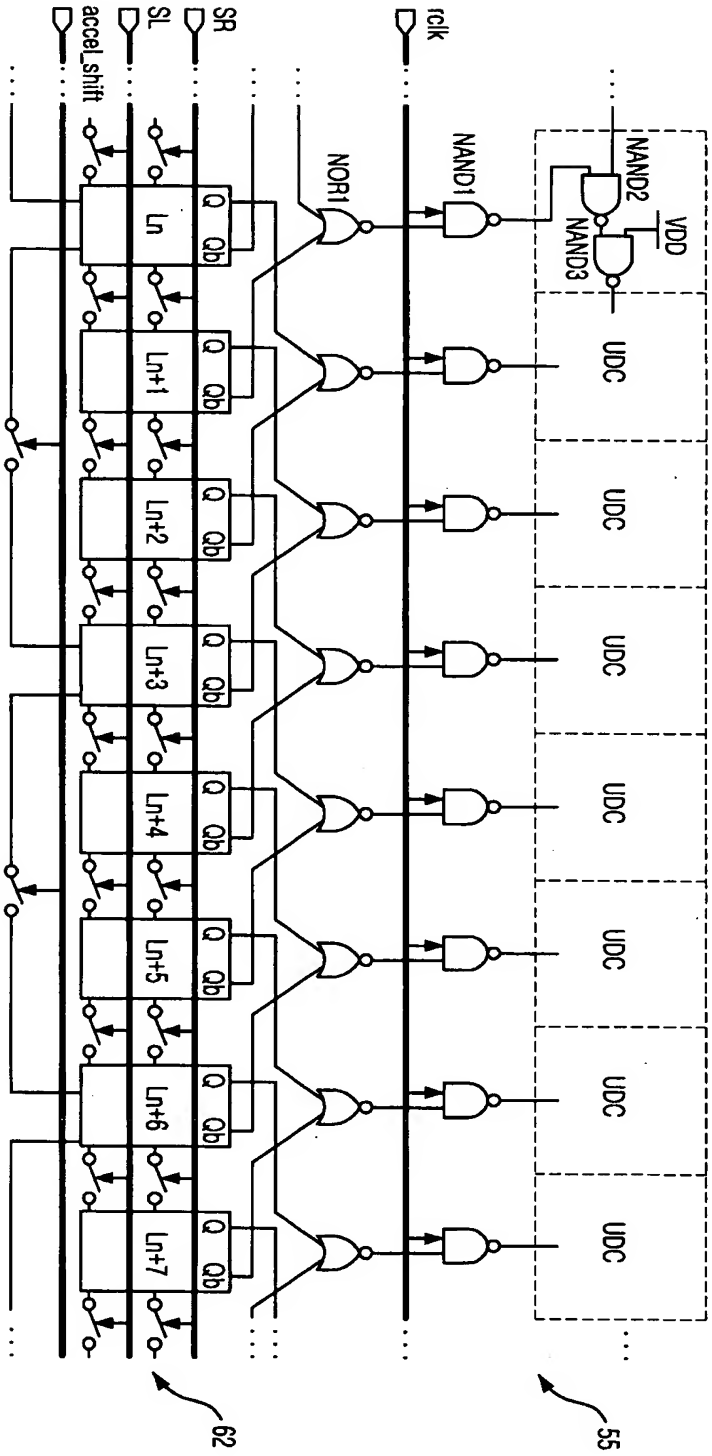
【도 4】


fb_dm
지연부
63
fbclk_dly
제2 위상 비교기
64
제1 위상 비교기
60
Pd2
Pd1
모드 결정부
65
accel_end
쉬프트 레지스터 제어부
61
SR
SL
accel_shift
쉬프트 레지스터
62
CLK
제2 클럭 입력 버퍼
52
rclk
/CLK
제1 클럭 입력 버퍼
51
fclk
제2 지연 라인
55
제1 지연 라인
54
rclk_dl
fclk_dl
지연 모델
59
제2 DLL 드라이버
58
제1 DLL 드라이버
57
rclk_dll
fclk_dll

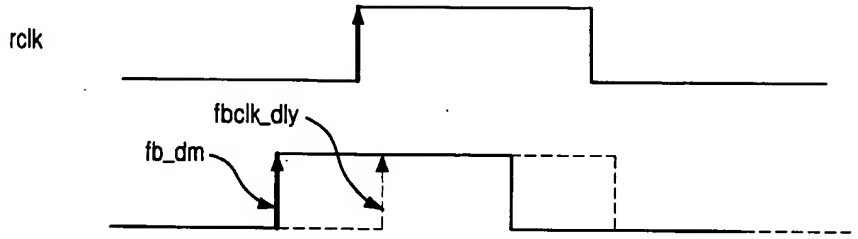
【도 5】


rclk
초기 동작시
fb_dm
fbclk_dly
> tTA
1번의 가속화 모드 동작 후
N x unit_delay
> tTA
2번의 가속화 모드 동작 후
> tTA
3번의 가속화 모드 동작 후

【도 6】

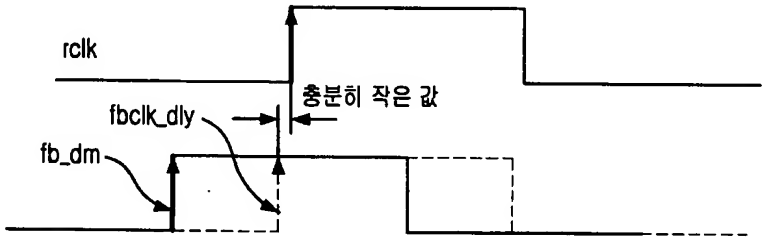

accel_shift
SL
SR
clk
NOR1
NAND1
NAND2
NAND3
VDD
Ln
Ln+1
Ln+2
Ln+3
Ln+4
Ln+5
Ln+6
Ln+7
Q
Qb
UDC
62
55

【도 7a】

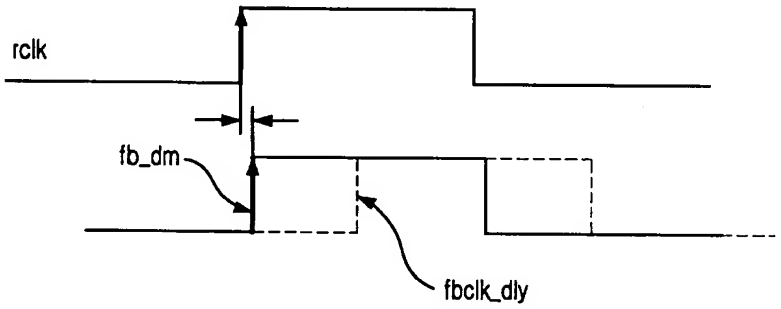


【도 7b】

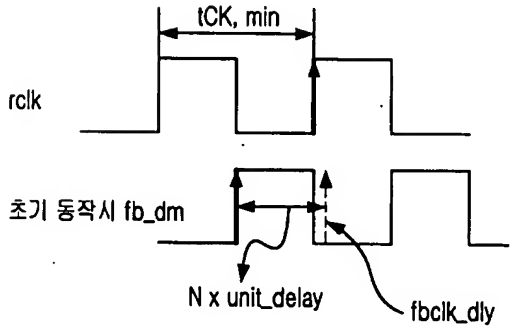
k번째 가속화 모드 동작 후



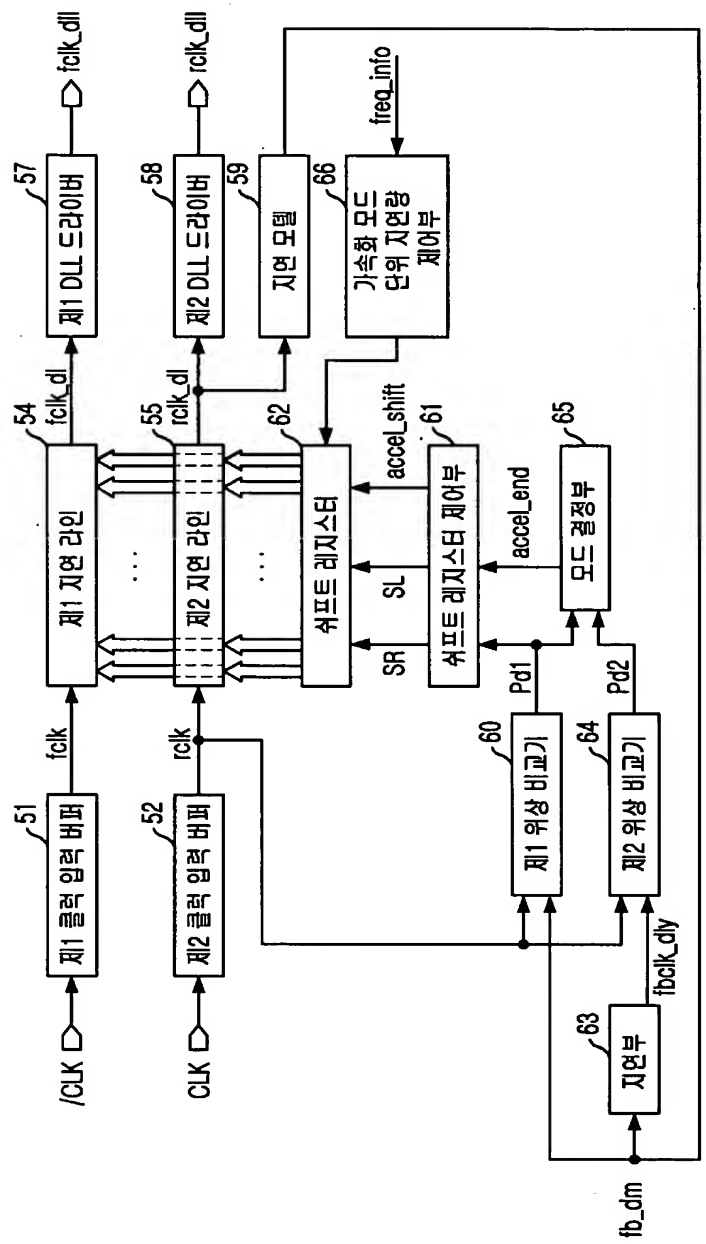
k+1번째 가속화 모드 동작 후



【도 8】



【도 9】


/CLK
CLK
fb_dm
51
제1 클럭 입력 버퍼
52
제2 클럭 입력 버퍼
fclk
rclk
54
제1 지연 라인
55
제2 지연 라인
fclk_dl
rclk_dl
57
제1 DLL 드라이버
58
제2 DLL 드라이버
fclk_dll
rclk_dll
59
지연 모델
62
쉬프트 레지스터
SR
SL
accel_shift
61
쉬프트 레지스터 제어부
66
가속화 모드 단위 지연량 제어부
freq_info
65
모드 결정부
accel_end
60
제1 위상 비교기
64
제2 위상 비교기
Pd1
Pd2
63
지연부
fbclk_dly